View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Scientific Repository

Implementation of Javanese Text to Speech using MaryTTS Engine

Rudy Adipranata[1], Yulia[2], Liliana[3], Gregorius Satia Budhi[4]
Informatics Dept
Petra Christian University
Surabaya, Indonesia
rudya@petra.ac.id[1], yulia@petra.ac.id[2], lilian@petra.ac.id[3], greg@petra.ac.id[4]

***Abstract*— Indonesia is a country consisting of many ethnic groups with each having different cultures and languages. One of the tribes in Indonesia is the Javanese, where most of the Javanese live on the island of Java. The Javanese have their culture along with its language and character. In this research, Javanese text to speech has been implemented, and as interface, word processor application has been developed. This word processor application is used to convert the writing in Roman character into Java character and also as input for the text to speech engine. MaryTTS has been used as text to speech engine for Javanese language. From experimental results, the accuracy of word processor for conversion reached 100% while the accuracy of Javanese text to speech reached 98.1%.**

Keywords- text to speech; Javanese language; MaryTTS

I. INTRODUCTION

As one of the Indonesia culture assets, Javanese culture and language needs to be preserved especially among the younger generation. Javanese language has its own unique form of letters called Java character and also it has pronunciation which differ from pronunciation of Indonesian language. This difference creates difficulty for people to read or write Javanese language literature or script. Several researches related to Java character have been done such as Java character recognition which is try to convert from Java character to Roman character [1,2,3,4] with aim to facilitate the learning of Java character easily especially for younger generation.

In this research, we implemented Javanese language text to speech. This can be used to learn the pronunciation of Javanese language. We used word processor application [5] as an interface and input for text to speech. This word processor application is used to convert from Roman character to Java character. In addition as interface for text to speech, by using this application, we can also use this to learn how to write Java character easily. As engine for text to speech, we used MaryTTS engine. MaryTTS engine is an open-source text to speech engine written in Java. It was originally developed as a collaborative project of DFKI's Language Technology Lab and the Institute of Phonetics at Saarland University [6].

The rest of this paper is organized as follows. In section II and III, we explain brief review about Java character, text to speech and MaryTTS. In section IV, we present the system design. Implementation and results are presented in section V. And in last section VI, we present our conclusion.

II. JAVA CHARACTER

Javanese language has special form for its character, called Java Character. Java character consists of 20 carakan characters, 20 pasangan characters and several sandhangan characters.

Carakan characters are core syllables that consist of 20 characters, which are called ha, na, ca, ra, ka, da, ta, sa, wa, la, pa, dha, ja, ya, nya, ma, ga, ba, tha, nga. The carakan characters can be seen in Table I [7].

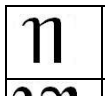
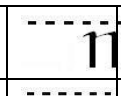
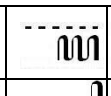
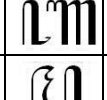
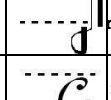
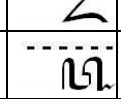
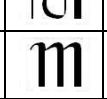
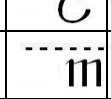
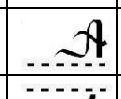
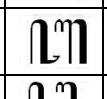
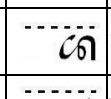
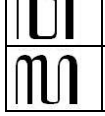
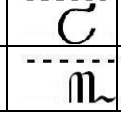
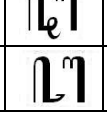
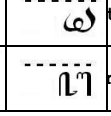
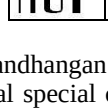
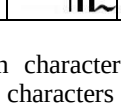
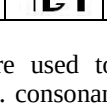
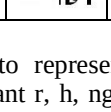
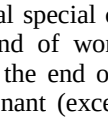
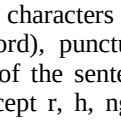
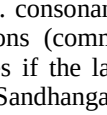
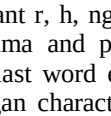
TABLE I. CARAKAN CHARACTERS

	Read		Read		Read		Read
	ha		da		pa		ma
	na		ta		dha		ga
	ca		sa		ja		ba
	ra		wa		ya		tha
	ka		la		nya		nga

Pasangan characters are used to represent consonant at the end of word, also consist of 20 characters where each carakan character has its own pasangan character. Some of pasangan should be written below carakan character and some of them should be written aligned with carakan character. The pasangan characters can be seen in Table II [7].

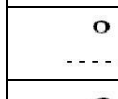
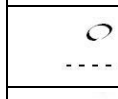
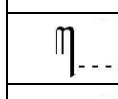
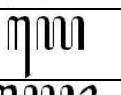
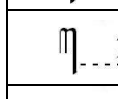
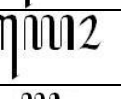
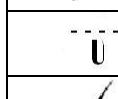
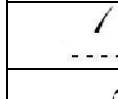
TABLE II. PASANGAN CHARACTERS

Carakan Char.	*Pasangan*	Read	*Carakan Char.*	*Pasangan*	Read
		ha			pa
		na			dha
		ca			ja

		ra			ya
		ka			nya
		da			ma
		ta			ga
		sa			ba
		wa			tha
		la			nga

Sandhangan characters are used to represent vowels, several special characters (e.g. consonant r, h, ng located at the end of word), punctuations (comma and period) and mark the end of the sentences if the last word ended with consonant (except r, h, ng). Sandhangan characters can be seen in Table III [7].

TABLE III. SANDHANGAN CHARACTERS

Symbol	Example	Read
		yi
		ye'
		ye
		yo
		yu
		yar
		yah
		yang
		k (consonant at the end of word)
		kra
		kre
		kya
		pa, (comma)
		pa. (period)

III. TEXT TO SPEECH AND MARYTTS

Text to speech is the process of converting text into voices. Speech synthesis is the process of making human speech artificially. Some text to speech engines use the machine learning method to produce sounds that are very similar to human voices. The engine must be trained with a lot of data in the form of text and audio, including words, sentences, and pronunciation [8].

A text to speech engine is usually divided into 2 major sections, front-end and back-end. The front-end will convert the text into a variety of meaningful data and information, ready for further processing for speech synthesis. The required data are pronunciation, intonation, speech, word and sentences. The back-end is assigned to run the speech sythesis process. It will receive input data from front-end, and its output is voice. Usually the back-end requires a lot of sound recording database, as a reference to produce the appropriate voice [9].

MaryTTS is an text to speech engine, consist of front-end and back-end. MaryTTS is designed to accept several types of inputs: plain text, SSML (Speech Synthesis Markup Language, a markup language based on XML for application of speech synthesis) and SABLE (another XML markup language to annotate texts for speech synthesis, but already discontinued in 2010) [6].

IV. SYSTEM DESIGN

In order to develop the word processor for Java character, we need to separate each syllable in words. The design for separation of syllable is using automata. The design results can be seen in Fig. 1 [10].

In the design in Fig. 1, vowel is represented by character V, consonant is represented by character C represents consonant, and the lowercase characters enclosed in quotation marks are input characters. At first, input come into the Q0 as initial state. Then this initial state will split into four subsequent states, Q2-Q5. If input is character n, state Q2 will be selected, if input is character d, state Q3 will be selected, if input is character t, state Q4 is selected and state Q5 is selected if input is character h, c, r, k, s, w, l, p, j, y, m, g, or b. The final states are Q6, Q10, Q12, Q13 and Q14 [10].

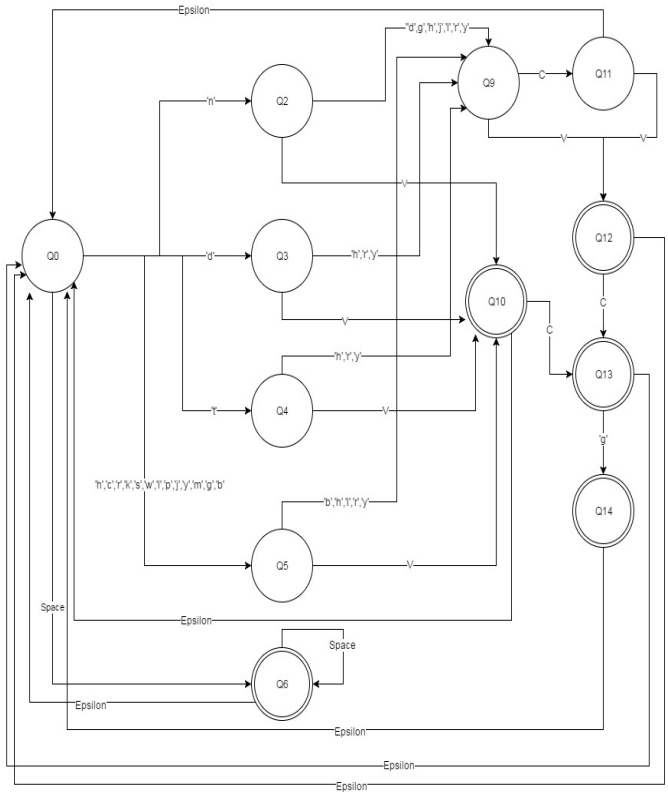


Figure 1. Design for syllable separation.

V. IMPLEMENTATION AND RESULTS

The first step for text to speech implementation using MaryTTS is create Javanese speech corpus. Speech corpus is database collection of voices along with text. In this research, the speech corpus was formed using approximately 1000 sentences of Javanese language. Each sentence has between 5 and 15 words. Some sentences can be seen in Fig. 2.

```
( jv_01_0001 "Ing jaman kawitan Gusti Allah nitahake langit lan bumi" )
( jv_01_0002 "Gusti Allah banjur ngandika: Anaa pepadhang Tumuli ana padhang" )
( jv_01_0003 "Padhang mau dipirsani wus becik lan Gusti banjur misahake padhang karo peteng"
)
( jv_01_0004 "Kang padhang diparingi aran raina, dene kang peteng: wengi" )
( jv_01_0005 "Mangkono wus dadi sore lan esuk: dina kang kapisan" )
( jv_01_0006 "Pepenthengan mau diparingi aran: langit" )
( jv_01_0007 "Mangkono wus dadi sore lan wus dadi esuk, dina kang kapindho" )
( jv_01_0008 "Kang asat kaparingan aran: bumi, lan kaklumpukane banyu diparingi aran:
segara" )
( jv_01_0009 "Kabeh mau dipirsani Gusti Allah wus prayoga" )
( jv_01_0010 "Mangkono wus dadi sore lan wus dadi esuk: dina kang katelu" )
( jv_01_0011 "Padha dadia pepadhang ana ing pepenthengan langit supaya madhangi bumi" )
( jv_01_0012 "Kaya mangkono wus dadi sore lan wus dadi esuk, dina kang kaping papat" )
( jv_01_0013 "Sarta iku dipirsani dening Gusti Allah wus prayoga" )
( jv_01_0014 "Tumuli padha diberkahi lan dipangandikani" )
```

Figure 2. Example of Javanese sentences.

The second step is create lexicon and allophone file. Lexicon is collection of words and their pronounciations. The pronounciations use in this lexicon using SAMPA notation. Example of lexicon in this research can be seen in Fig. 3.

```
adeg a_d-@\g
adhedhasar a_-d`e-d`a_-s`a_r`
adhegan a_-d`@\-ga_n`
adhem a_-d`@\m
adhi a_-d`i
adhinira a_-d`i-n`i-r`O
adhopsi a_-d`Op-s`i
adhung a_-d`uN
adi a_-di
adidung a_-di-duN
adigang a_-di-ga_N
adiguno a_-di-gu-n`o
adil a_-dil`
adinegoro a_-d`i-n`@\-go-r`o
adipati a_-di-pa_-ti
adisucipto a_-di-s`u-tSip-to
adiwacara a_-di-wO-tSO-r`O
```

Figure 3. Example of Javanese lexicon.

After create lexicon, also we need to create allophone file using XML format. This allophone file contain all phonemes in Javanese language. All phonemes used in lexicon creation, must be the same as phonemes listed in allophones XML. There should be no phoneme in lexicon that is not listed in allophones XML. The lexicon and allophones are processed to become language pack later use by MaryTTS engine.

Besides language pack, it also needs to build voice pack using speech corpus that have been made in first step. There are two ways to build voice, by unit selection or HMM based. Unit selection will result better speech synthesis, more like human voice but need a lot of speech corpus and the long time training process. HMM based requires shorter processing time, use less speech corpus but the result of speech synthesis is not as good as unit selection, but still resembles human voice. In this research, we used HMM based to build voice pack.

After doing all the above steps, MaryTTS engine is ready for use by word processor application. This word processor application is the extended version from word processor application that has been developed in previous research [2]. In the last research, it used its own hanacaraka font, while currently the application used Google Noto Sans Javanese font. This application is also developed so that text written on word processor can communicate with MaryTTS engine for voicing the text. In this word processor, the user only uses a regular keyboard (Roman character) to write a sentence and the application will automatically display the Java character in accordance with the sentence. The conversion is done in accordance with the design that has been made. The word processor application can be seen in Fig. 4.

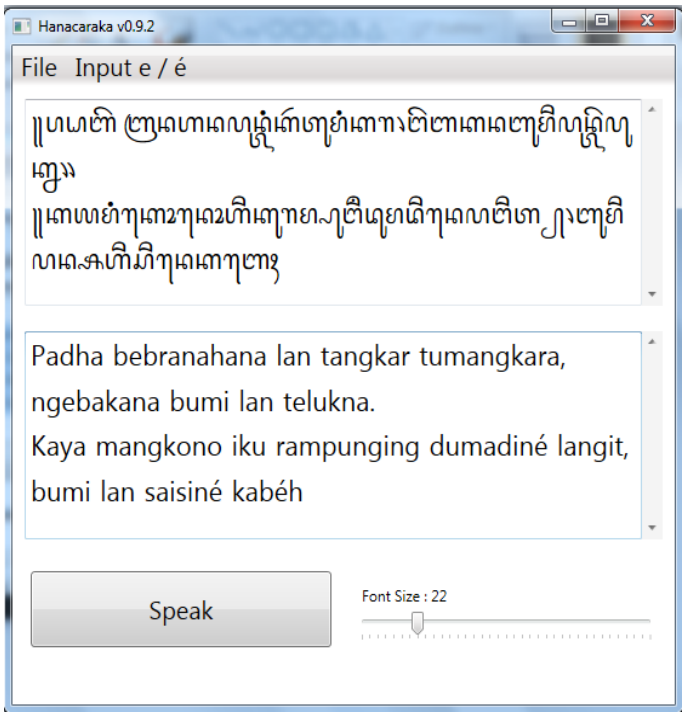


Figure 4. Word processor and text to speech application.

To measure the conversion accuracy of word processor and text to speech applications, about 700 sentences is used. From the experiment result, the accuracy of the conversion from Roman to Java character is 100%, while the accuracy of text to speech is about 98.1%. The errors of text to speech occur especially in some words that have the same character, but the pronunciation is different.

VI. CONCLUSION

This research has implemented Javanese text to speech along with word processor application as its interface. The word processor application can do conversion from Roman character into Java character. From the experimental results, the accuracy of word processor conversion reached 100%, and the accuracy of text to speech reach 98.1%. The results of this research can further be used for interactive learning of speech and writing Javanese character.

ACKNOWLEDGMENT

This research was funded by DIPA Directorate General of Research and Development Reinforcement (Direktorat Jenderal Penguatan Riset dan Pengembangan) no. 120/SP2H/LT/DRPM/2018, fiscal year 2018. We also thank Christopher Imantaka Halim for his help in this research.

REFERENCES

[1] Adipranata, R., Liliana, Indrawijaya, M., Budhi, G.S., "Feature extraction for Java character recognition," Communications in Computer and Information Science 516, 2015, pp. 278-288.

[2] Budhi, G.S., Adipranata, R., "Handwritten javanese character recognition using several artificial neural network methods," Journal of ICT Research and Applications vol.8, no.3, 2015, pp. 195-212.

[3] Budhi, G. S., Adipranata, R., "Comparison of bidirectional associative memory, counterpropagation and evolutionary neural network for Java characters recognition," Proceeding of The 1st International Conference on Advance Informatics: Concepts, Theory and Applications, Bandung, Indonesia, 2014, pp. 7-10.

[4] Budhi, G. S., Adipranata, R., "Java characters recognition using evolutionary neural network and combination of chi2 and backpropagation neural network," International Journal of Applied Engineering Research, vol 9, no 22, 2014, pp. 18025-18036.

[5] Adipranata, R., Budhi, G. S., & Thedjakusuma, R., "Java characters word processing," Proceeding of The 3rd International Conference on Soft Computing, Intelligent System and Information Technology, Bali, Indonesia, 2012.

[6] MaryTTS, "The Mary text-to-speech system (MaryTTS)", Retrieved 3 Jan 2017 from http://mary.dfki.de.

[7] Tofani, Abi & Nugraha, Setyo, Tatanan anyar pinter basa jawi pepak, Pustaka Agung Harapan, Surabaya.

[8] Rubin, P., Baer, T., Mermelstein, P., "An articulatory synthesizer for perceptual research," Journal of the Acoustical Society of America. 70 (2), 1981, pp. 321–328.

[9] Van Santen, Jan P. H., Sproat, Richard W., Olive, Joseph P., Hirschberg, Julia, Progress in speech synthesis. Springer, 1997.

[10] Yulia, Liliana, Adipranata, R., Budhi, G.S., "Design of Javanese text to speech application," Journal of Telecommunication, Electronic and Computer Engineering, in press.